グレブナー基底計算のための weight生成アルゴリズム

AUTHOR(S):

木村, 欣司; 野呂, 正行

CITATION:

木村, 欣司 ...[et al]. グレブナー基底計算のためのweight生成アルゴリズム. 数理解析研究所講究録 2004, 1395: 1-7

ISSUE DATE:

2004-10

URL:

http://hdl.handle.net/2433/25919

数理解析研究所講究録 1395 巻 2004 年 1-7

グレブナー基底計算のためのweight生成アルゴリズム

木村欣司
KINJI KIMURA
神戸大学自然科学研究科
GRADUATE SCHOOL OF SCIENCE AND TECHNOLOGY, KOBE UNIVERSITY *
野呂正行
MASAYUKI NORO
神戸大学理学部数学科
DEPARTMENT OF MATHEMATICS, KOBE UNIVERSITY †

1 はじめに

この講究録では, グレブナー基底計算のための weight 生成アルゴリズムについて述べる. グレブナー基底計算は, 数学, 物理, 工学において重要な道具である. グレブナー基底計算の計算戦略は, ユーザーによって設定される項順序と weight に依存する. そして, 計算の複雑度はそれらに大きく影響される. 項順序を自動的に生成する試みはすでに行われており, 我々は weight についても自動的に生成することを試みた. ここでは, 効果的な weight を自動的に生成するいくつかの方法を提案し実際の問題に適応してそれらの有効性を検討する.

2 概要

完全に斉次な多項式系のグレブナー基底の計算は, 次数を基準とした選択戦略のもとでよい振る舞いをすることが経験的に知られている. よい振る舞いの典型例として, 係数膨張を起こすことが少ないことを挙げる. さらには, 入力された多項式系が斉次でなくとも weight を設定することで斉次化できるならばその weight のもとでグレブナー基底は効率的に計算される. そのような weight は, 入力された多項式系の exponent vector より構成される線型方程式を解くことで得られる. たとえ, 多項式系を斉次化する weight が存在しなくともなんらかの weight がグレブナー基底の計算に効果的に働くことがよく観られる. 我々は, 次の経験則を持っている. 与えられた多項式系の Newton polytopes をより"slender"にできるならば, 短い時間でグレブナー基底を計算できる. 我々はこの経験則に沿った weight を構成するさまざまな方法で定式化をおこない, その経験則において最適であるとおもわれる weight を与えるアルゴリズムを提案する. そして, 実際の問題からそのアルゴリズムで weight を構成しそれを用いた時の有効性を検討する.

*kimura@math.koba-u.ac.jp
†noro@math.koba-u.ac.jp

3 断り

この講究録では, floating の計算を避けるように定式化する. その理由は, 分野の違いから我々が floating 計算に精通しているとは言い難いからである. しかし, その後の研究で floating の計算を必要とする定式化のほうがよい weight を生成することがわかった. floating の計算を必要とする定式化は, "Risa/Asir Journal" に投稿する予定である.

4 weight によって斉次化される場合

$$P = \{tu^{10} + stu^9 + s^2tu^8 + t^3u^7 + s^3tu^7, tu^{11} + stu^{10}, tu^{12} + stu^{11}\}$$

を例にとる. ここで, weight

$$w = (s, t, u) = (2, 3, 2)$$

を定める. これは, s に 2,t に 2,u に 2 を weight として与えることを意味する. すると, 多項式集合 (1) は共通の weight(1) によって完全に斉次化される. このような w を見つけるには以下の手順をとる.

一般に,

$$P = \{f_1, ..., f_m\}, \quad f_i = \sum_{j=1}^{L_i} c_{ij} x^{e_{ij}} \quad (i = 1, ..., m). \tag{1}$$

多項式集合が与えられたとき, P を斉次化する共通の weight w は存在するならば次の式を満たす.

$$we_{i1} = we_{i2} = \cdots = we_{iL_i} \quad (i = 1, ..., m). \tag{2}$$

(2) は,

$$w(e_{i2} - e_{i1}) = 0, ..., w(e_{iL_i} - e_{iL_{i-1}}) = 0 \quad (i = 1, ..., m) \tag{3}$$

と同値であり連立線形方程式系 (3) を得る. 概して (3) は過剰決定系であり incremantal procedure を用いて効率よく解くことができる:

1. 連立線形方程式系から適当な subsystem を選びそれを解く.
2. 解が存在しないならば系全体にも解は存在しない.
3. 解が一意的に決まるならば残った方程式にその解を代入しそれが系全体の解であることを確かめる.
4. さもなければ, subsystem に新たな式を追加しこの procedure を繰り返す.
5. 多項式系の解が次元をもつならば, 不等式系を解くことでその次元をもった解から適当に点を選ぶ. weight は正数要素のみを許すためである.

例として神戸大学の佐々木先生の問題をあげる.

$$\begin{aligned} P &= \{f_1, f_2, f_3\}, \\ f_1 &= h_1c_1c_1 + 2h_2c_1c_2 + 2h_3c_1c_3 + h_4c_2c_2 + 2h_7c_2c_3 \end{aligned}$$

$$
\begin{aligned}
& -(h_1+h_4)c_3c_3 - 2m_1c_1 - 2m_2c_2 - 2m_3c_3, \\
f_2 \;=\; & h_2c_1c_1 + 2h_4c_1c_2 + 2h_7c_1c_3 + h_5c_2c_2 + 2h_6c_2c_3 \\
& -(h_2+h_5)c_3c_3 - 2m_2c_1 - 2m_4c_2 - 2m_5c_3, \\
f_3 \;=\; & h_3c_1c_1 + 2h_7c_1c_2 - 2(h_1+h_4)c_1c_3 + h_6c_2c_2 \\
& -2(h_2+h_5)c_2c_3 - (h_3+h_6)c_3c_3 - 2m_3c_1 - 2m_5c_2 + 2(m_1+m_4)c_3.
\end{aligned}
$$

線形方程式系を解くと, パラメータ (m,h) を含んだ解を得られる:

$$w = (c_1,\ldots,c_3,m_1,\ldots,m_5,h_1,\ldots,h_7), \tag{4}$$

c_i, m_i, h_i は,

$$c_i = m-h \quad (i=1,\cdots,3), \quad m_i = m \quad (i=1,\cdots,5), \quad h_i = h \quad (i=1,\cdots,7).$$

正数要素の weight を得るため $(m,h)=(2,1)$ を選ぶ:

$$w = (1,\ldots,1,2,\ldots,2,1,\ldots,1). \tag{5}$$

5 いかなる weight によっても斉次化されない場合

前の section の線型方程式系が解を持たないならば, 多項式集合はいかなる weight によっても斉次化することはできない。入力多項式集合がある weight と項順序によりすでにグレブナー基底になっている場合には, それを見つけるアルゴリズムはすでに与えられている.[1] しかし, 入力多項式集合がグレブナー基底でない場合, 最適な weight を見つけるアルゴリズムは見つかっていない. この講究録で, 我々は設定しないときより計算時間が短くなる 3 つの実験的アルゴリズムを提案する. 我々の指導原理は, weight によって与えられた多項式集合をできる限り斉次化された系に近づけるようにすることである. 我々は, この方針を次のように定式化する:

はじめに, 我々は入力多項式集合の exponent vector から線型方程式系 $Ax=b$ をつくる. ここにおいて, A は $m \times n$ の行列, x は n 次元のベクトル, b は n 次元のベクトルである. 次に, 我々は $\|Ax-b\|_2$ を最小化する x を見つけるため最小 2 乗法をもちいる.

最小 2 乗法は, つぎの性質に基づいている,

$$\|Ax_0-b\|_2 = \min\|Ax-b\|_2 \Leftrightarrow A^TAx_0 = A^Tb, \tag{6}$$

ここにおいて, 右辺は正規方程式とよばれる. すべての成分が正の最小 2 乗解 x が得られたならば, 我々はそれを weight としてもちいる. この正規方程式を効率的に解くため fraction free Gaussian 法を用いる.

例として, 次の多項式集合からはじめる,

$$P = \{tu^{11}+tu^{10}+stu^9+s^2tu^8+t^3u^7, tu^{11}+stu^{10}, tu^{12}+stu^{10}\}, \tag{7}$$

この多項式集合には, はじめの多項式集合 (1) とほぼ同じような項が現れている. 我々は, 方程式系 $Ax=b$ を得るために 3 つの戦略を提案する.

5.1 各々の多項式に異なったtotal degreeの目標を与える戦略

多項式集合 (7) から次の式を得る,

$$A=\begin{pmatrix}0&1&11&-1&0&0\\0&1&10&-1&0&0\\1&1&9&-1&0&0\\2&1&8&-1&0&0\\0&3&7&-1&0&0\\0&1&11&0&-1&0\\1&1&10&0&-1&0\\0&1&12&0&0&-1\\1&1&10&0&0&-1\end{pmatrix},x=\begin{pmatrix}s\\t\\u\\t_1\\t_2\\t_3\end{pmatrix},b=\begin{pmatrix}0\\0\\0\\0\\0\\0\\0\\0\\0\end{pmatrix},Ax=b. \tag{8}$$

多項式集合 (7) が weight によって斉次化されるならば, この系は最適な weight を与える. 方程式系 (8) に最小 2 乗法を適用し性質 (6) を用いると, 方程式系 (8) から次の式を得る:

$$A^TAx_0=A^Tb. \tag{9}$$

解は,

$$x_0=(0,0,0,0,0,0), \tag{10}$$

であり明らかに不適である.

この困難を避けるため 1 変数を正規化する. 例えば,

$$s=1. \tag{11}$$

この正規化のもとで, 式 (8) は次のように変形される,

$$C=\begin{pmatrix}1&11&-1&0&0\\1&10&-1&0&0\\1&9&-1&0&0\\1&8&-1&0&0\\3&7&-1&0&0\\1&11&0&-1&0\\1&10&0&-1&0\\1&12&0&0&-1\\1&10&0&0&-1\end{pmatrix},y=\begin{pmatrix}t\\u\\t_1\\t_2\\t_3\end{pmatrix},d=\begin{pmatrix}0\\0\\-1\\-2\\0\\0\\-1\\0\\-1\end{pmatrix},Cy=d. \tag{12}$$

方程式系 (12) に最小 2 乗法を適用し性質 (6) を用いると, 方程式系 (12) から次の式を得る:

$$C^TCy_0=C^Td. \tag{13}$$

解は,

$$y_0=(29/24,2/3,199/24,209/24,217/24), \tag{14}$$

$$w=(s,t,u)=(1,29/24,2/3). \tag{15}$$

同じ方向を向いた二つの weight は等価であり, かつ大きな正数は計算によい影響を及ぼすことは少ない. よって, 次の丸めの戦略を用いて丸める.

- strategy 1

  得られた weight vector の最小の要素を 1 に規格化し, 残った要素は最も近い正の整数に丸める.

- strategy 2

  得られた weight vector に対して最小の成分を 1 に規格化した後適当な正の整数を乗算しすべての成分が整数に十分近くなるようにする. その後, 規格化されたもの以外の成分を最も近い正の整数に丸める.

この例では, strategy 1 から $w=(2,2,1)$ を strategy 2 から $w=(9,11,6)$ を得る.
同じ計算を他の規格化についてもおこなう,

$$t=1 \quad \text{or} \quad u=1 \quad \text{or} \quad t_1=1 \quad \text{or} \quad t_2=1 \quad \text{or} \quad t_3=1. \tag{16}$$

このようにたくさんの最適な weight の候補を得られてしまうのがこの定式化の問題点である. その問題を避けるには floating の計算を必要とする定式化を行えばよいのであるが, その詳細については"Risa/Asir Journal"に投稿する予定である.

5.2 すべての多項式に共通の total degree の目標を与える戦略

多項式集合 (7) から, 次の式を得る,

$$M=\begin{pmatrix} 0 & 1 & 11 \\ 0 & 1 & 10 \\ 1 & 1 & 9 \\ 2 & 1 & 8 \\ 0 & 3 & 7 \\ 0 & 1 & 11 \\ 1 & 1 & 10 \\ 0 & 1 & 12 \\ 1 & 1 & 10 \end{pmatrix}, z=\begin{pmatrix} s \\ t \\ u \end{pmatrix}, n=\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}, Mz=n, \tag{17}$$

これは, 式 (8) に次の条件を課すことで得られる,

$$t_1=t_2=t_3=1. \tag{18}$$

方程式系 (17) に最小 2 乗法を適用し性質 (6) を用いると, 方程式系 (17) から次の式を得る:

$$M^TMz_0=M^Tn. \tag{19}$$

解は,

$$z_0=(s,t,u)=(2430/21367, 3403/21367, 1618/21367), \tag{20}$$

である. Section 5.1 の戦略を使って $(2,2,1)$ あるいは $(12,17,8)$ に丸められる.

5.3 重複したexponent vectorを省略した後すべての多項式に共通のtotal degreeの目標を与える戦略

多項式集合 (7) から, 次の式を得る,

$$R=\begin{pmatrix}0&1&11\\0&1&10\\1&1&9\\2&1&8\\0&3&7\\1&1&10\\0&1&12\end{pmatrix},p=\begin{pmatrix}s\\t\\u\end{pmatrix},q=\begin{pmatrix}1\\1\\1\\1\\1\\1\\1\end{pmatrix},Rp=q, \tag{21}$$

ここで注目していただきたいのは, $(0,1,11)$ と $(1,1,10)$ が一度だけ現れていることである. この場合, 同じ exponent vectors は一度だけ数えることとする.

方程式系 (21) に最小2乗法を適用し性質 (6) を用いると, 方程式系 (21) から次の式を得る:

$$R^TRp_0=R^Tq. \tag{22}$$

解は,

$$p_0=(s,t,u)=(324/2741,435/2741,208/2741), \tag{23}$$

である. Section 5.1 の戦略を使って $(2,2,1)$ あるいは $(3,4,2)$ に丸められる.

6 項順序の決定

ここまで, 効率的なグレブナー基底計算のための weight を決定する方法について議論してきた. 項順序を決定する際にも, weight を考慮すべきという考えから項順序を決定する方法についても述べる. 再び, 経験則を述べる:

weight の重い順に変数を並べて the weighted total degree reverse lexicographic ordering を用いて計算するならば, 効率よくグレブナー基底を計算できる. もし weight について各々の変数がまったく異なった値をもつならば, lexicographic ordering を使ったほうがよい.

もしまったく同じ weight を持つ変数が存在したならば, ユーザーに従うこととする.

7 Timing Data

この section では, さまざまな setting における実際の問題のグレブナー基底計算の timing data を紹介する. 次の表を説明する. *Sawada* は, 沢田先生の方法による項順序のみの提案を採用することを意味する. 特に, 特別な weight を設定することはしない. *W-homo* は, 入力多項式を Section 4 による方法で得られた weight によって完全に斉次化できたことを意味する. *Method 1,2,3* は, それぞれ Section 5.1, 5.2, 5.3 に対応した方法を示す. *Strategy 1,2* は, それぞれ Section 5.1 で示した丸めの戦略を採用したことを示す. *Sawada* 以外の項順序は, Section 6 で示した方法によって決定する. weight を設定した後, 斉次系でないものには1変数増やして斉次化し trace lifting を適用する. その際, 増やした1変数の weight は1に設定する. *Method 1* では, すべての変数を一度1に規格化し best と worst の時間を記載する.

7.1 weightによって斉次化される場合

Fujimoto's problem [3]

	weight vector	term ordering	time
Sawada	not set	[q,u,w,e,k,o,d,c,f,b,a], DRL	out of memory (2G byte)
W-homo	[20, 18, 14, 12, 8, 6, 5, 4, 3, 2, 2]	[w,u,q,o,k,e,d,c,b,f,a], LEX	254 seconds + 3 hours

この系は, weightによって斉次化されたった254秒でグレブナー基底の候補が生成できたことを示す. 候補が真のグレブナー基底であるためのcheckに3時間を必要とする. しかし, Fujimotoさんの問題はradical membership problemすなわち与えられた多項式をいくつかのべき乗したものがグレブナー基底の候補によって0に還元されることを確かめる問題である. 与えられた多項式の2乗がグレブナー基底の候補によって0に還元されるのを確かめることは簡単でcheckのための3時間は必要ない.

7.2 いかなるweightによっても斉次化されない場合

McKay's problem [4]

	weight vector	term ordering	time
Sawada	not set	[a5,a3,a2,a1], DRL	26280 seconds
Method 1, best	[28, 20, 16, 11]	[a5,a3,a2,a1], DRL	2932 seconds
Method 1, worst	[40, 34, 23, 12]	[a2,a5,a3,a1], DRL	> 26280 seconds
Method 2, Strategy 1	[2, 2, 1, 1]	[a5,a3,a2,a1], DRL	12630 seconds
Method 2, Strategy 2	[37, 28, 22, 16]	[a5,a3,a2,a1], DRL	6497 seconds
Method 3, Strategy 1	[3, 2, 2, 1]	[a5,a3,a2,a1], DRL	4960 seconds
Method 3, Strategy 2	[19, 13, 11, 7]	[a5,a3,a2,a1], DRL	2931 seconds

計算の過程を解析してみると, McKay's problemではweightを設定しない場合と比較すると係数の膨張が起きていないことがわかった. それが計算時間を大幅に短縮している.

はじめ, 与えられた多項式系のNewton polytopesを"slender"にすることでreductionのstep数を減少させることができると考えていた. しかし, すくなくともMcKay's problemの場合にはたしかに係数膨張は押さえられるがreductionのstep数は増加していた. 一般にこの二つは相反することである. ゆえに我々のweightを決める方法はheuristicの段階にあり理論的なsupportは存在しない. にもかかわらず, 我々のweightはこれ以外の多くの問題でグレブナー基底の計算コストを減らす効果がみられるので今後もこの研究を続けていきたい. なお, ここでは紙面の関係で2つの例のみを載せることをお詫びする.

参 考 文 献

[1] P.Fritzmann and B.Strumfels(1993), Minkowski addition of polytopes: computational complexity and applications to Gröbner bases, SIAM Journal of Discrete Mathematics 6,246-269

[2] 沢田浩之, グレブナ基底計算を効率的に行うための項順序自動設定法. 数式処理 Vol.9 No.2 (2002), 56-77.

[3] The system of equations is available from `http://www.math.sci.kobe-u.ac.jp/HOME/kimura/fujimoto`

[4] M. Noro and J. McKay, Computation of replicable functions on Risa/Asir. Proceedings of the Second International Symposyum on Symbolic Computation PASCO'97, ACM Press, 130-138 (1997). The system of equations is available from `http://fgbrs.lip6.fr/jcf/Benchs/@benchs/mckay.fgb`.